|  |  |  |
| --- | --- | --- |
| **Versión** | **Fecha** | **Cambios** |
| V4 | 10-7-2025 | Versión inicial |
| V5 | 21-7-2025 | Apéndice I: Centralización de configuración de acceso a Oracle. |
| V6 | 31-7-2025 | Añadido en subrayado amarillo al principio del Apéndice i una leyenda que especifica que el partado sólo afecta a micros que usan el conector sql |
| V7 | 7-8-2025 | Añadidas las variables del Config Map que hay que quitar (sólo para DevOps) |

**Decomisado Seguro de mainframe en aplicaciones Darwin:**

**Eliminación de elementos obsoletos Gravity**

**Contexto**

Tras la adopción del modo Only Gravity España, la configuración sw que soportó la ejecución en dual-run de microservicios (mainframe / Gravity) deja de tener sentido.

Este documento es una guía para eliminar de forma segura propiedades o valores innecesarios, con configuración más simples para el escenario final Only Gravity.

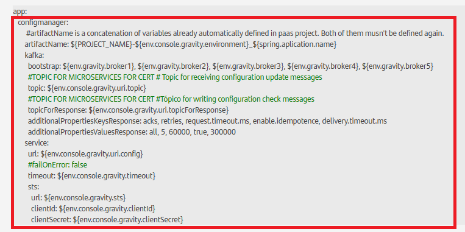
Las propiedades / valores de configuración a revisar son:

* **Las del Configuration Manager**, debido a que esté elemento será deprecado al perder todo el sentido su existencia (ya no hay agujas que cambiar; ha quedado todo en Only Gravity).
* **Las del modo de ejecución** en cada uno de los conectores trx, sat o sql, como consecuencia de que el escenario final Only Gravity queda establecido de forma fija y ya no habrá un Configuration Manager que les especifique a las aplicaciones en qué modo deben ejecutar cada conector (trx, sat o sql).
* **Las de los endpoints de conexión a mainframe**, ya que al apagar mainframe los hosts / puertos CICS serán decomisados.
* **Las del pool de conexiones a DB2**, ya que sólo quedará el Oracle de Gravity.
* **Las de los tópicos de monitorización,** ya que algunos de ellos sólo eran necesarios para trazar / comparar en fase de dual-run.

1. **Propiedades del Configuration Manager (CM)**

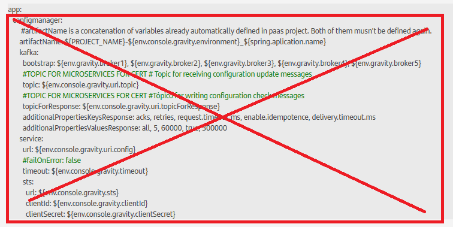
El CM fue el elemento clave durante el dual-run para establecer el modo de ejecución del micro en arranque, (llamada al servicio securizado por un token sts) o cambiar las agujas en caliente (conexión a tópicos kafka de escucha y respuesta).

**Situación de partida.** Microcon las propiedades del CM para cambiar de agujas en arranque o en caliente.

****

**Acciones a realizar**: Eliminar completamente el bloque de props *app.configmanager*

**Situación final:** Micro sin el bloque de propiedades *app.configmanager*



1. **Propiedades del modo de ejecución**

Durante el dual-run han prevalecido los valores de modo de ejecución dual para las propiedades de los conectores trx, sat y sql.

Estos modos de ejecución han estado en el caso de versiones Darwin superiores a 3.2.3 siempre establecidos por el CM.

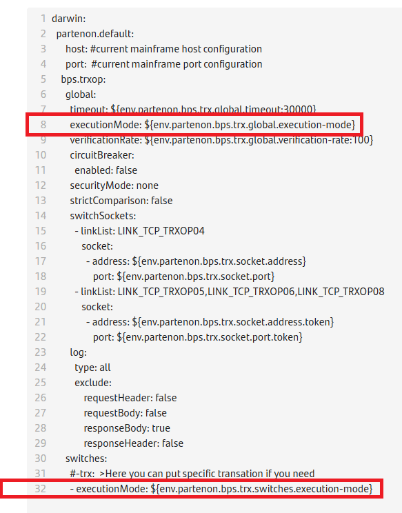
Por tanto, lo que se va a hacer en estos casos es fijar las propiedades a Only Gravity fijas en el yml porque ya no hay CM.

Este cambio a valores fijos Only Gravity sin CM hay que hacerlo diferente para cada conector:

* **Propiedades *executionMode* en conector trx**: puede haber más de una, dependiendo de si hay configuraciones propias para cada canal, para cada trx -op-versión con referencias o para trx con enql o Hades.
* **Propiedades *executionMode*** (o *execution-mode*) **en conector sat**: puede haber más de una si hay trx con referencias.
* **Propiedades option en conector sql**.

**Situación de partida:** Micro que utiliza uno o varios conectores y que está en modo OnlyMainframe o dual run master mainframe porque luego recibe (en arranque o en caliente la orden del CM para cambiar)

a) Conector Trx



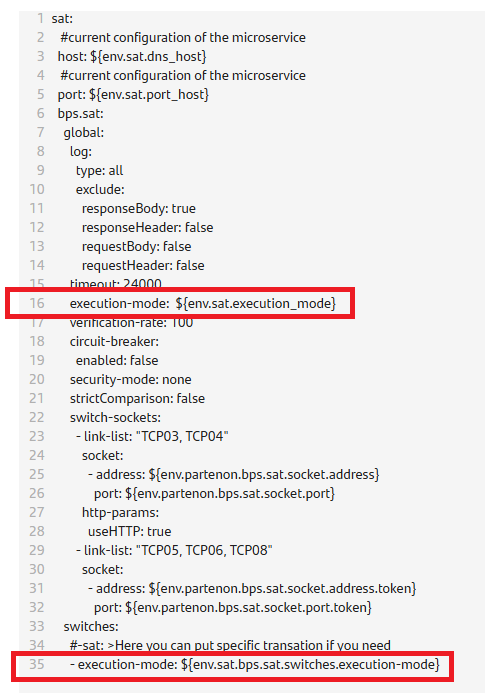
Se parte de la propiedad *darwin.partenon.bps.trxop.global.executionMode* puesta a valor *test* lo cual establece dual-run con master mainframe. (línea 8 del ejemplo)

Ojo es posible que no esté bajo el epígrafe d*arwin* dado que se puede haber montado el conector trx-op directamente sin utilizar el wrapper Darwin para dicho conector.

P*ero para obedecer al CM en los cambios de aguja*, fue necesario poner la propiedad  *darwin.partenon.bps.trxop.switches.executionMode* a valor *disabled*  lo cual desactiva el switch para que mande el CM (línea 32 de este ejemplo).

Puede haber varias ocurrencias, dependiendo de si hay configuraciones específicas por canal o incluso en cada uno de ellos haber debajo del switches hay configuraciones específicas para cada trx por referencias o enq2l o Hades.

b) Conector Sat



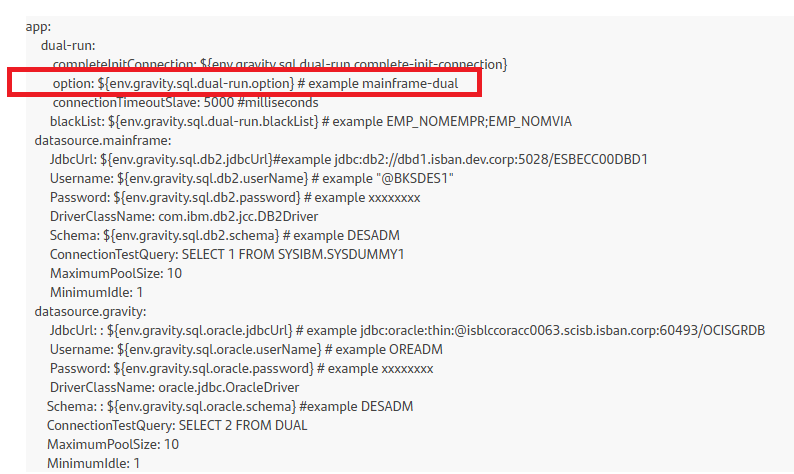
Se parte de la propiedad sat.bps.sat.global.execution-mode puesta a valor *test* lo cual establece dual-run con master mainframe. (línea 16 del ejemplo).

A veces, también la nomenclatura de la variable, puede ser *executionMode* que es equivalente a *execution-mode.*

*Pero para obedecer al CM en los cambios de aguja*, fue necesario poner la propiedad sat.bps.sat.switches.execution-mode   a valor *disabled*  lo cual desactiva el switch para que mande el CM (línea 35 de este ejemplo).

Puede haber varias ocurrencias, dependiendo de si hay configuraciones específicas por uso de referencias en algunos de los sats.

c) Conector Sql (Mds)



Se parte de la propiedad app.dual-run.option puesta a valor *mainframe-dual* o *mainframe*.

Este conector no necesita ninguna propiedad más para el CM porque entra “de oficio”

**Acciones a realizar y resultado final**

a) **Conector trx**. Se trata de forzar los *executionMode (o execution-mode)* de la secciones global y switches a una combinación de valores que fuerza el modo Only Gravity en la variable del apartado global con el valor *enabled* impidiendo que entre a mandar el CM poniendo también a *enabled* el valor de las variable/s del apartado swiches.

Para ello, se deben repasar los siguientes valores y fijarlos a enabled

darwin.partenon.bps.trxop.global.executionMode : enabled

darwin.partenon.bps.trxop.switches.executionMode : enabled

+ (por cada configuración de canal) executionMode : enabled

+ (por cada referencia) executionMode : enabled

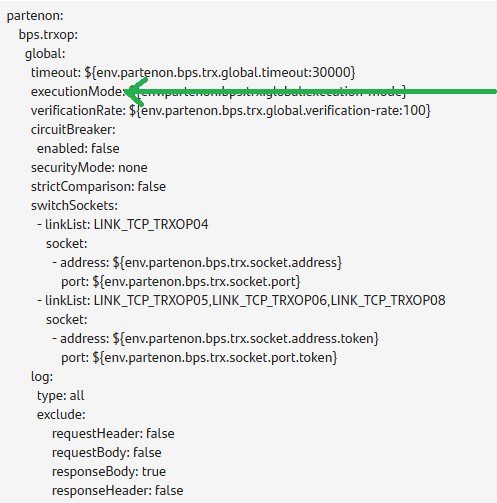
+ (si eq2l) darwin.partenon.bps.trxop.global.reentry : enabled

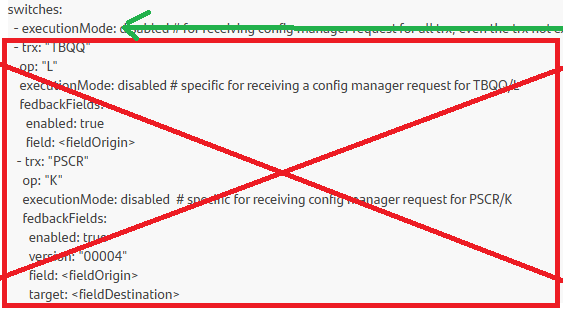
+ (si eq2l) en cada trx con eq2l reentry : enabled manteniendo para esa trx executionMode : verify

(\*) si no se usa wrapper Darwin, todas estas variables sin prefijo “darwin.”

La estructura del yml permanece exactamente igual, salvo en el caso de referencias, se puede aprovechar para quitar completamente las secciones de las referencias, dado que al no haber dual-run ya no tienen sentido realimentar campos a un secundario.

El ejemplo muestra un caso con una referencia que se elimina completamente (aspa roja) y los valores del executionMode de las secciones switches y global **(líneas verdes) que deben ir a *enabled***.





b) **Conector sat**. Se trata de forzar los *executionMode (o execution-mode)* de la secciones global y switches a una combinación de valores que fuerza el modo Only Gravity en la variable del apartado global con el valor *enabled* impidiendo que entre a mandar el CM poniendo también a *enabled* el valor de las variable/s del apartado swiches.

Para ello, se deben repasar los siguientes valores y fijarlos a enabled

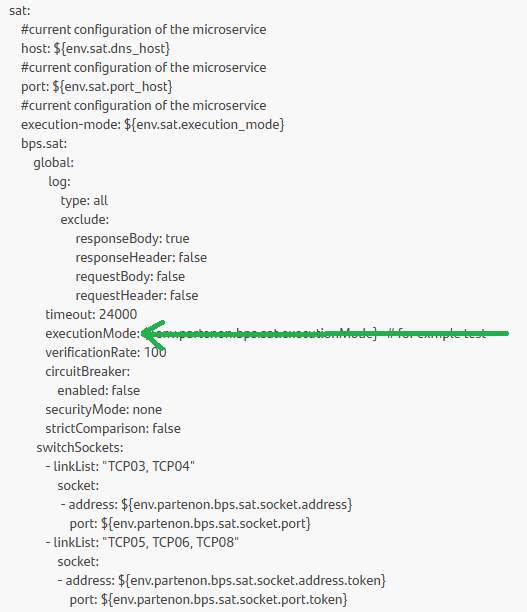
sat.bps.sat.global.execution-mode : enabled

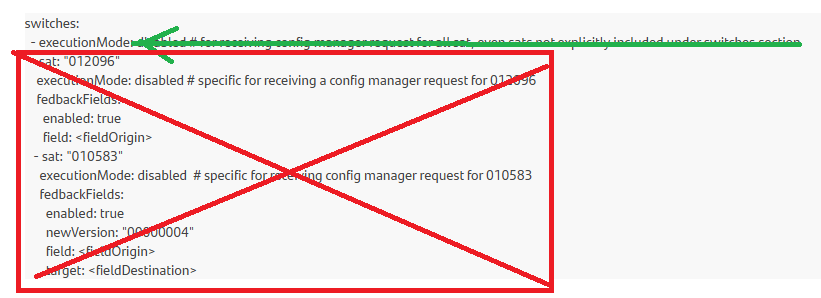
sat.bps.sat.switches.execution-mode : enabled

+ (si hay referencias) cada execution-mode : enabled

La estructura del yml permanece exactamente igual, salvo que en el caso de referencias, se puede aprovechar para quitar completamente las secciones de las referencias, dado que al no haber dual-run ya no tienen sentido realimentar campos a un secundario.

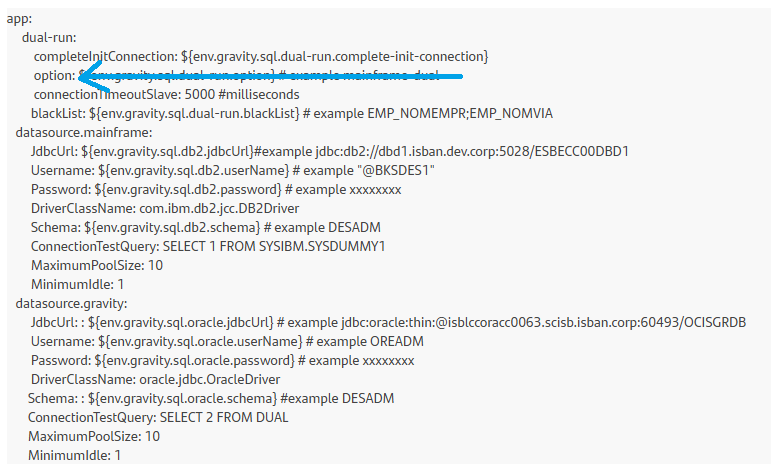
El ejemplo muestra un caso con una referencia que se elimina completamente (aspa roja) y los valores del execution-mode de las secciones switches y global **(líneas verdes) que deben ir a *enabled***.





c) **Conector sql (mds)**: Se trata de forzar el parámetro *option* a valor *gravity*

Para ello, se debe poner el valor app.dual-run.option : gravity (línea azul del ejemplo)

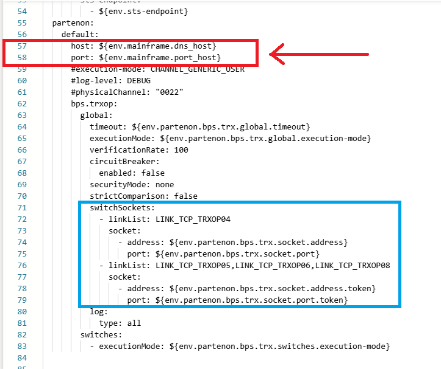


1. **Propiedades de los endpoints de conexión a mainframe**

Aunque hay más de una forma de hacerlo, se ha determinado que la forma más segura consiste en mantener la estructura de los ymls tal y como están y sustituir los valores de las variables que especifican los endpoints (host y puerto) del antiguo mainframe / DB2.

**Situación de partida:** Micro que utiliza uno o varios conectores que tienen propiedades apuntando a host / puerto de las regiones mainframe para trx o sat o que tienen configurado un datasource contra DB2 mainframe o alguno de los UDBs.

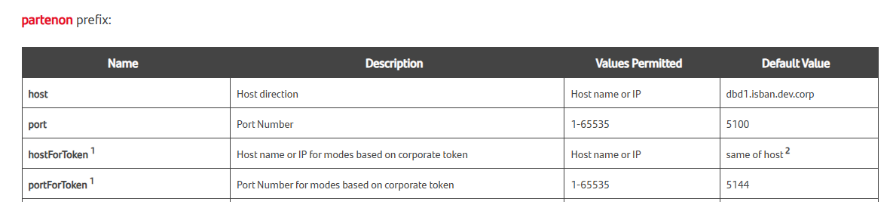
a) Conector trx:



Se parte de un micro que tiene configuradas las propiedades host y *port* del mainframe (líneas 57 y 58 del ejemplo) marcadas en rojo. Este host y puerto van a ser deprecados.

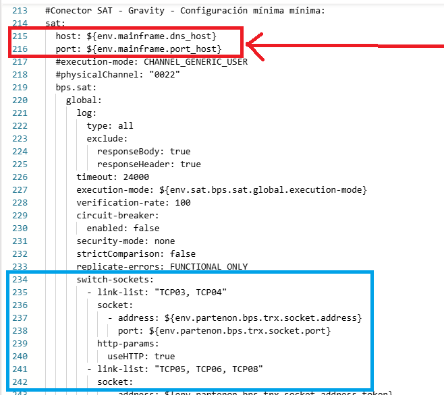
Diferénciensen estas propiedades de las de la sección switchSockets (líneas 71 a 79 del ejemplo marcadas en azul) que son los endpoints de Gravity y los que deben mantenerse.

Adicionalmente, puede tener al mismo nivel las propiedades *hostForToken* y *portForToken*



E incluso podría tener más de una vez estas propiedades si hay cfg. para cada canal.

b) Conector sat



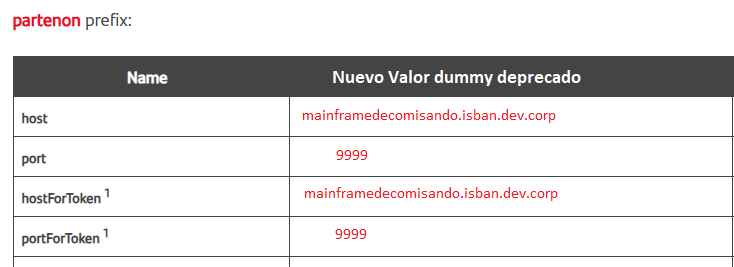
Se parte de un micro que tiene configuradas las propiedades host y *port* del mainframe (líneas 215 y 216 del ejemplo) marcadas en rojo. Estos host puerto van a ser deprecados.

Diferénciensen estas propiedades de las de la sección switchSockets (líneas 234 a 243 del ejemplo marcadas en azul) que son los endpoints de Gravity que deben mantenerse.

**Acciones a realizar y resultado final**

Las acciones a realizar en estos casos consisten en localizar las propiedades *host, port, hostForToken* y *portForToken* allá donde aparezcan en los conectores trx y sat y sustituirlas por un host y un puerto dummy, pero las secciones de los ymls se mantienen exactamente tal y como están.

1. Conector trx

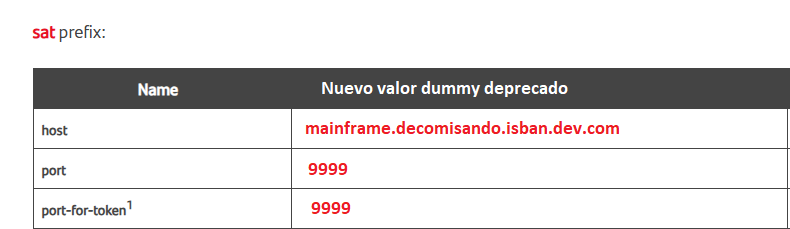


El yml se mantiene exactamente igual. Sólo cambiar los valores de estas propiedades en donde aparezcan.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

1. Conector sat



El yml se mantiene exactamente igual. Sólo cambiar los valores de estas propiedades en donde aparezcan.

Interfaz de usuario gráfica

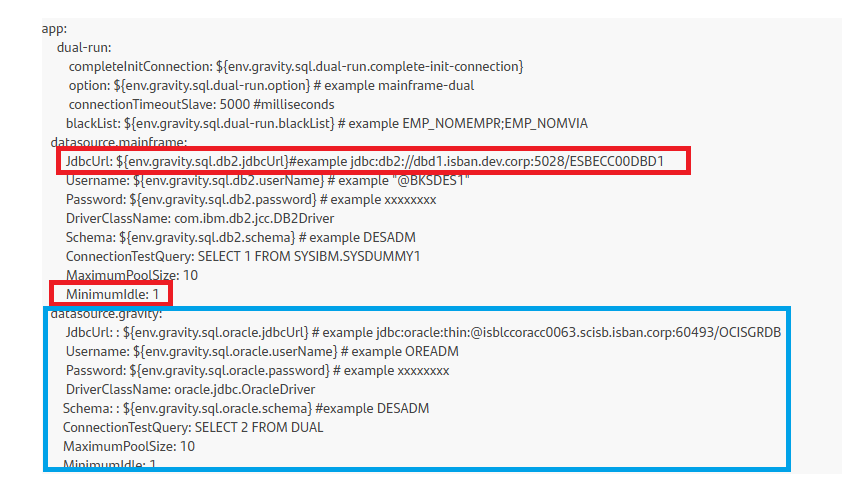
El contenido generado por IA puede ser incorrecto.

1. **Propiedades del pool de conexiones a DB2**

Dado que el DB2 mainframe y los UDBs también serán decomisados, las configuraciones deben quedar claras para eliminar o mockear las referencias.

El conector MDS se mantiene tal cual, con las mismas propiedades

**Situación de partida**

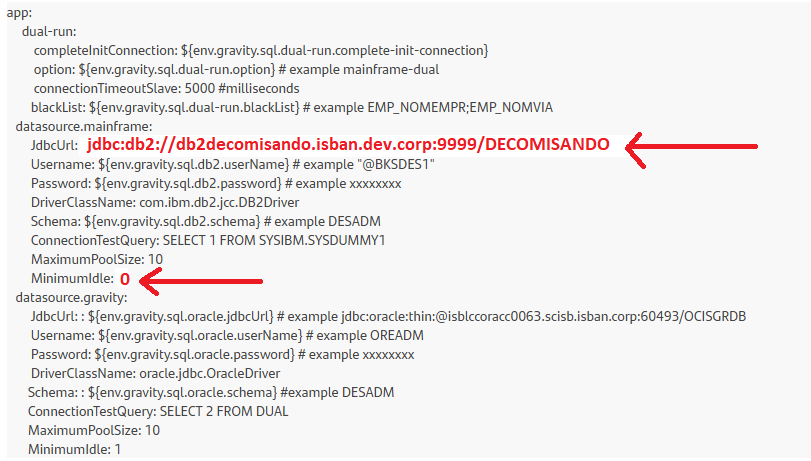
****

Se parte de un micro configurado con los datasources del conector MDS (normalmente 2 el de DB2 mainframe y el del Oracle Gravity, aunque puede haber casos en que haya también otras bases de datos Oracle o DB2 UDBs).

Los elementos a decomisar son precisamente las bases de datos DB2 mainframe y UDB, especialmente las propiedades marcadas en rojo: *jdbcUrl* con la cadena de conexión y el tamaño mínimo del pool de conexiones *MinimumIdle.*

**Acciones a realizar y resultado final**

Dado que se han deprecado, se pondrá un dummy en la cadena de conexión al DB2 y se establecerá que el número mínimo de conexiones sea cero para evitar que en arranque del micro intente levantar el pool a la base de datos mockeada y deje error en las trazas.



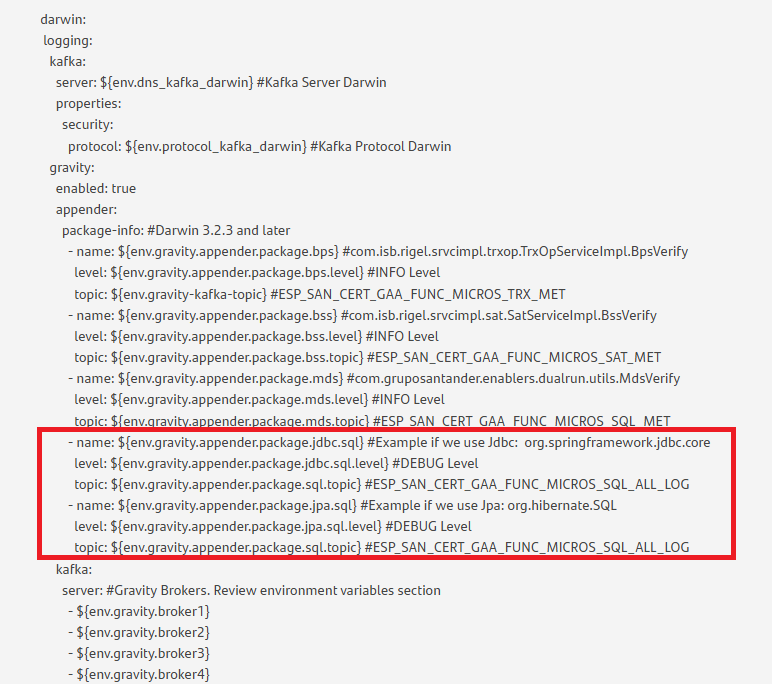
Nótese que el yml permanece exactamente igual. Sólo cambiaron estos valores.

1. **Propiedades de los tópicos de monitorización**

Es decisión post-proyecto mantener la monitorización de trx, sat y sql en modo Only Gravity.

Sin embargo, la monitorización del del sql-funcional debe ser eliminada. Por tanto, en caso de que se esté monitorizando sql-funcional todos aquellos tópicos que monitoricen el tráfico de las clases org.springframework.jdbc.core y/o org.hibernate.SQL deben ser quitados de la configuración (o subir su nivel a ERROR para que no aparezcan o poner un nombre de tópico dummy, ej ESP\_SAN\_CERT\_GAA\_FUNC\_MICROS\_SQL\_ALL\_LOG\_DEPRECADO).

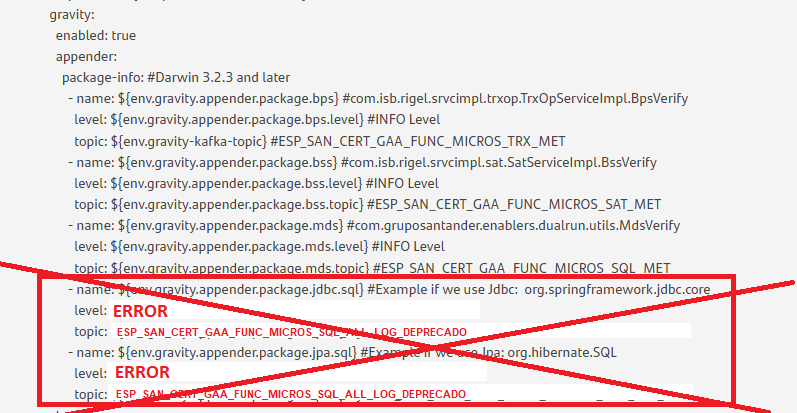
**Situación de partida**

****

Se parte de un micro en el se han configurado a máximos todos los tópicos posibles.

**Acciones a realizar y resultado final**

Se deben eliminar los appenders del sql-fun o inutilizarlos



Es decir, sólo se mantienen los bloques de las clases *BpsVerify*, *BssVerify* y *MdsVerify*.

Y se eliminan los que haya de sql-funcional que a máximos pueden ser estos 3: *org.springframewokk.jdbc.core*, *org.hibernate.SQL* y *com.gruposantander.enablers.dualrun.logs.MdsActivityLog*

**Adicionalmente el equipo de Devops realizará esta misma acción en en el ConfigMap** **Gravity** en el que se introdujeron estas mismas propiedades para el Kafka seguro.   
  
Eliminar las siguientes variables:

APP\_CONFIGMANAGER\_KAFKA\_BOOTSTRAP

APP\_CONFIGMANAGER\_KAFKA\_TOPIC

APP\_CONFIGMANAGER\_KAFKA\_TOPICFORRESPONSE

DARWIN\_LOGGING\_GRAVITY\_APPENDER\_PACKAGEINFO\_3\_TOPIC

DARWIN\_LOGGING\_GRAVITY\_APPENDER\_PACKAGEINFO\_4\_TOPIC

DARWIN\_LOGGING\_GRAVITY\_APPENDER\_PACKAGEINFO\_5\_TOPIC

DARWIN\_LOGGING\_GRAVITY\_APPENDER\_PACKAGEINFO\_5\_NAME

DARWIN\_LOGGING\_GRAVITY\_APPENDER\_PACKAGEINFO\_4\_NAME

DARWIN\_LOGGING\_GRAVITY\_APPENDER\_PACKAGEINFO\_3\_NAME

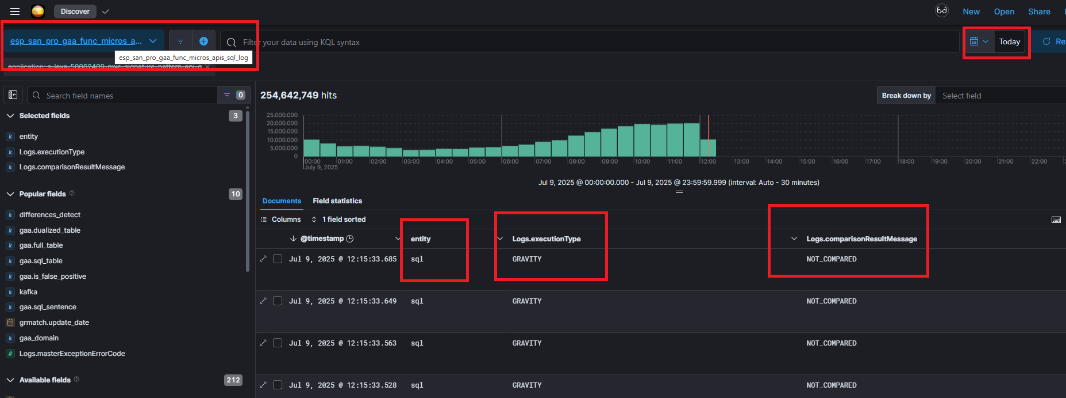
1. **Evidencia de funcionamiento.- Prueba final y baja definitiva en el CM**

La prueba de que estos cambios son correctos implica:

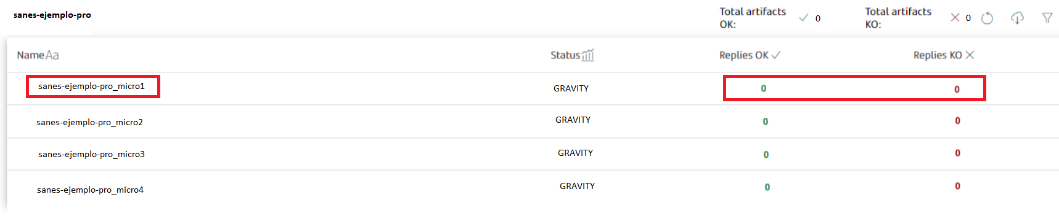
1. **Hacer prueba mínima regresiva** de la aplicación para ver que sigue funcionando y generando tráfico de trx, sat y sql si los hubieren.
2. **Examinar la traza en kibana** para ver que se siguen viendo se las escrituras en el índice y dichas escrituras evidencian que el micro se mantiene ejecutando en modo Only Gravity.
   1. **Trx / Sat:** Mirar el tópico trx / sat y revisar los campos*: Logs.disptachWay* (que debe tener típicamente el valor *SOCKET\_SWITCH*) y *Logs.comparisonResult* (que debe tener típicamente el valor *NOT\_COMPARED*). Ejemplo de PRO:



* 1. **Sql:** Mirar el tópico sql y revisar los campos*: Logs.executionType* (que debe tener típicamente el valor *GRAVITY*) y *Logs.comparisonResultMessage* (que debe tener típicamente el valor *NOT\_COMPARED*). Ejemplo de PRO.



1. **Examinar la traza en Dynatrace** (o traza en fichero)para comprobar que ya no hay ninguna llamada al Config Manager y que los modos de ejecución Only Gravity que se ven en Kibana vienen determinados por el valor de las propiedades del micro y no por las llamadas al Config Manager
   1. **CERT:** *<https://config-manager-gateway.santander.dev.corp/config-manager/config>*
   2. **PRE:** *https://config-manager-gateway.santander.pru.bsch/config-manager/config*
   3. **PRO:** *<https://config-manager-gateway.santander.corp.bsch/config-manager/config>*
2. **Comprobar reply la consola del CM:** Asegurar que ningún pod del entorno probado hace reply(en el caso de OCU, diferenciar los pod de los de PRO).

****

1. **Solicitar la baja definitiva en el CM**: Una vez que se comprueba que ya ningún pod hace replies (y junto con la baja solicitar a devops que quite también el check de conexión previo de conexión al CM).
2. **Apéndice I: Centralización de configuraciones de acceso a Oracle**

**MUY IMPORTANTE: Este cambio afecta sólo a micros que tienen acceso a sql. Si el micro sólo tiene trx, no hay que hacer nada.**

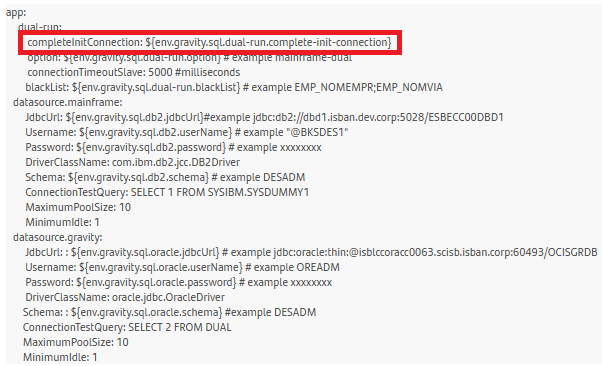
Posteriormente a la implantación de Gravity en PRO, los expertos Oracle han determinado que debe ajustarse la parametría de las sesiones Oracle para optimizar el rendimiento de la Base de Datos.

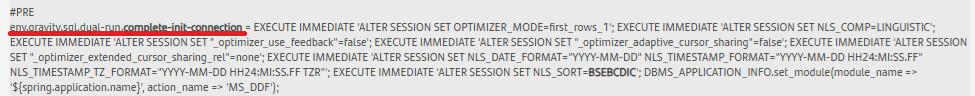
Esta parametría se establece al crear los pooles de conexiones.

* A nivel trx y sat de forma totalmente transparente a los micros, puesto que se gestiona a nivel producto Microfocus. Luego nada se debe tocar en el micro.
* Pero a nivel sql es el propio micro el que la gestiona a través de la propiedad del conector *app.dualrun.completeInitConnection*. Luego debe modificarse el valor de dicha propiedad

**Situación de partida:**

Se parte de un micro con la propiedad del conector *app.dualrun.completeInitConnection* fijada (normalmente, si se ha seguido la paáctica recomendada al valor de una variable *${env.gravity.sql.dual-run.complete-init-connection}* que luego se especifica el properties de cada entorno).





**Acciones a realizar y resultado final**

La propiedad en el yml se mantiene tal cual, pero se debe sustituir el valor de la variable en el properties por un nuevo que es el que recomiendan los expertos en Oracle

env.gravity.sql.dual-run.complete-init-connection: ADMTOOLS.GRAV\_SESSION\_DDF\_MICROS (pModuleName => '${spring.application.name}');

Solamente este cambio.

Obsérvese que este nuevo valor simplifica el que había anteriormente porque incluye únicamente una referencia a un procedimiento almacenado.

Dicho procedimiento ha sido creado por los expertos Oracle para incluir en él los detalles de la sesión Oracle y las mejoras de rendimiento en un punto único centralizado, de forma que las optimizaciones futuras ya no requerirán redesplegar los micros, ya que con este cambio quedan desacoplados para siempre micros y propiedades de sesión Oracle.